企業情報システムの理想像
－サイロフリーでのプログラミングレスを実現する－

はじめに

Ⅰ　サイロ(SILO)フリー指向のシステムアーキテクチャ
Ⅰ．１　情報処理空間（ＰＤ）とその拡大
Ⅰ．２　３階層通信場アーキテクチャ
Ⅰ．３　ビジネスシステムの構造とＥ２通信場
Ⅰ．４　企業間通信場Ｅ３の考察
Ⅰ．５　マネジメント通信場Ｍ２
Ⅰ．６　ＤＨアーキテクチャ事例
Ⅰ．７　システムアーキテクチャ構築の手順
Ⅰ．８　６種類のＤＨタイプ
Ⅰ．９　今後の課題

Ⅱ　図面によるシステムの客観的可視化
Ⅱ．１　図面言語
Ⅱ．２　ＴＨデータモデル
Ⅱ．３　ＩＰＦチャート
Ⅱ．４　メタデータ定義
Ⅱ．５　加工データの扱い
Ⅱ．６　データ意味論
Ⅱ．７　個別アプリの開発手順
Ⅱ．８　ザックマンアーキテクチャと椿の方法論

Ⅲ　プログラミングレスの方法
Ⅲ．１　プログラミングレスの狙い
Ⅲ．２．ＩＯデータモデル
Ⅲ．２．１　データ機能ｃと連携関数
Ⅲ．２．２　ＩＯ区画の導入
Ⅲ．２．３　見出しと属性値の配置
Ⅲ．２．４　処理手順
Ⅲ．３　処理系ミドルウエアの構造

Ⅳ．事例研究

Ⅳ．１　購買契約（Ｊ１）
Ⅳ．２　発注（Ｊ２）
Ⅳ．３　時間割（Ｊ３）

おわりに

はじめに

コンピュータが導入されてから、ほぼ５０年が、またＲＤＢMS（Relational Data Base Management System）が導入されてからは、３０年が経つ。ＥＸＣＥＬ、スマホ、ｉＰＡＤ、インターネット、クラウド、とＩＴの進化は素晴らしい。しかし、みずほ銀行のトラブルに限らず、企業情報システムの運用・開発に伴うトラブルは減らない。丸投げ開発によるブラックボックス・システムやアプリケーション・パッケージが乱立して、M&Aや広域システム連携のニーズに俊敏に対応できない。情報システムコストは増大し、情報システム部門は３K職場として、人気が下降している。

何かが間違っている。情報処理の機械化によって、生産性アップを図った。最新の情報技術の導入を心掛けてきた。しかし最終的なビジネスのあるべき姿、これを支える情報システムのあるべき姿をどう描いたのだろうか。情報システム開発は建設業の一種である。都庁の建設には、精密な図面が書かれ、発注者、設計者、施工業者がこれを共有したはずである。見えない情報システムの建設にどのような図面が書かれ、共有されたのだろうか。レントゲンもＭＲＩも無いまま、患者を診断する医者のようになってはいないだろうか。「ポンチ絵と物語」でコミュニケーションする手工業でなく、個人差の出ない図面によりコミュニケーションする近代的エンジニアリングに進化させなければならないはずと考えるのだが。

また開発された情報システムからは、種々の画面・帳票（ＩＯ）が製品として作られる。しかし、多くの企業において、どのような画面・帳票が作られているのか、そのリストが簡単には用意できない。製造業、たとえば自動車会社に行って、製品リストが用意されていないことは考えられないが、情報システム部門では、その製品、画面帳票のリストが入手できなくても、それは当たり前のことになっている。

企業のアプリケーション・システムは、通常孤立したものではない。他のシステムと何らかのメッセージ――SEやプログラマーがトランザクションと呼ぶイベントデータ――のやり取りを行っている。しかしそれはシステム開発が終わってから配慮されることが多い。このため、当該システムが、これを含むより大きなシステム、たとえば全社システムに、どう位置づけられるかが、開発担当者の頭の中に画かれていない。これは最近、「孤島システム」とか「SILO システム」として問題視されているが、建物ができてから、都市計画を考えるようなことになる。本来あるべき姿の共有されないまま、当面の重点課題の解決だけを念頭に近視眼的に取組む、「外部設計無しの、内部設計先行アプローチ」である。こうして、「複雑なコードの変換を行わなければつながらない」、「システム間のやりとりがスパゲッティ状になる」など複雑化を招き、終わりの見えない維持管理が続く。

そこでＹ２Ｋにも有効と、企業全体を視野に入れたＥＲＰパッケージが登場したが、これですんなり解決できる企業は例外的であり、多くはＥＲＰパッケージをカスタマイズし、既存のレガシーシステムや他のアプリケーション・パッケージと組み合わせて運用せざるを得ない。画面や取扱説明書から理解するＥＲＰパッケージや個別業務パッケージは、可視化には程遠く、ブラックボックスを含んだまま担当者が世代交代すれば、システムのブラックボックス化は一層進行する。ＥＳＢ（Enterprise Service Bus）やＳＯＡ（Service Oriented Architecture）の提案もあるが、実装ツール主導の対症療法であり、可視化は進まず抜本的な対策とはなりにくい。

「情報システムの役割は経営の支援であるべき」とされているから、これは本来、限定された領域の実装を担当するＳＩベンダーの課題ではない。ＳＩベンダーとしては、自らの売上減となる可能性のある課題には積極的には取り組めない。全社ビジネスの合理化のために、これを可視化できるのは、自社の人材であり、その育成抜きで、これが実現できるはずもない。ビジネスとは役割・組織に分かれて、画面・帳票（情報）をやり取りし、顧客に有効な財（商品・サービス・情報やお金）を提供して、対価を頂く仕組みである。コンピュータやプログラム登場以前からの、いわゆる実装独立の仕組みであり、その最小単位は画面・帳票を構成するデータ――正確には属性・属性値――である。

システムの不整合・不具合のほとんどは、このデータやデータ相互間の値や形式の不整合、またタイミングに起因するから、これを可視化し、調整・標準化する必要がある。これを個別アプリケーション内だけでなく、情報を共有するアプリケーション横断、さらに本来は、企業横断にまで拡大して考えなければならない。これは企業横断のリソースデータ（通称マスターデータ）の標準化／ＭＤＭ（Master Data Management）およびメタデータ（顧客コード、顧客住所といった属性のデータタイプや所属などの仕様等）の標準化、正しい関連付けによって達成されるから、データ・アーキテクチャ・モデルを求めて、更なるＤＯＡ（Data Oriented Architecture）の推進を行わなければならないと、私は考える。

Ⅰ．１～Ⅰ．９で述べるように、社内の標準を共有するデータ空間（ＤＨ：Data Hub）を、リソースデータ、メタデータのほかに、Ｅ１（各個別アプリケーション）、Ｅ２（アプリケーション間）、Ｅ３（企業間）、Ｍ２（経営・アプリケーション間）の４種に分けて考察したが、「従来後回しとなっていたＥ２、Ｍ２を最優先で可視化するのが、最も効率的である」との結論を得た。Ｅ２、Ｍ２は安定しており、これをシステム関係者全員が共有するとき、Ｅ１見直しの優先度、見直しの戦略が正しく、かつ効率的に決定できる。また、リソースデータすら、Ｅ２、Ｍ２のモデリング時に、合わせて行うのが最も効率的と考えるに至った。そしてさらに、ＤＨの特徴からは、Ⅰ．８のように６種類に分類することが出来るこ

とがわかった。こうして従来扱い方のはっきりしなかったＤＨを、データモデルの一部として、最上位に位置づけて扱うことになった。

ＤＯＡとはまた、ユーザのアプリケーション要件を、処理要件とデータ要件に分離して記述しようとするものであったから、プログラミングレスは当初からの目標であった。比較的単純なケースについては、種々成功例も報告されているが、全社スコープ、さらに企業横断スコープでの配慮はなく、また複雑な意思決定データの入力を支援する高度なユーザニーズに対応するには到っていない。

そこで、これまで４０年間蓄積してきたＤＯＡノウハウを発展させ、企業横断のスコープでのプログラミングレスを実現する方法論を、Ⅲ．１～Ⅲ．３で考察し、Ⅳ．１～Ⅳ．３で事例研究する。狙いは当然、開発・維持の生産性アップもあるが、プログラムの果たすべき役割の本質を明らかにし、情報システムの理想像を明らかにすることの方が、より重要のようにも思われる。

なお、本書のベースとするＤＯＡは実装独立のＤＯＡである。そのドキュメンテーションは、実装環境が変化しても変化せず、ビジネスモデルの変化によってのみ変化するものである。実装環境は、予想以上に変化する。したがって、実装独立に不備のあるシステムは、実装環境とビジネスモデルの両方の変化に対応せざるを得ず、メンテ地獄に陥る。システム関係者の多くは、これまで［作って何ぼだ］という造り屋ＳＥ指向から、実装環境と抱き合わせで、システム問題に取組んできたため、実装独立の発想が出来ない。かなりのＯＪＴトレーニングが必要であるように見える。

また、われわれの言うＤＯＡは、データと処理の交点にある画面・帳票／ＩＯをベースに置くＤＯＡである。決してＲＤＢにストアするデータのみを対象にするＤＯＡではない。その意味では、ＩＯＤＡ（IO Driven Approach）というべきかも知れない。ＩＰＦ（Information Process Flow）チャートによって、必要十分のＩＯを洗い出し、このＩＯを構成するデータ／属性によって、まずＴＨ（Tsubaki-Hotaka）データモデルＩＯ部分図をつくり、次にこれを統合して、統合図をつくる。こうしてユーザの必要とするすべてのデータから成るデータモデル図ができる。

必要十分のＩＯを確定するためのＩＰＦチャートおよびＴＨデータモデルが、ここに述べるＩＯＤＡを支える２本柱と言える。方法論とはこのように、なるべくドキュメントの少ない上流工程でビジネスニーズを確定し、後はこれを着実に正しく実現していくべきものと考える。

この２つの図法、ＩＰＦチャートおよびＴＨデータモデル図については、Ⅱ．１～Ⅱ．５に簡単な解説をつける。これらは、本書の主題「通信場モデル」の解説を先行させるため、Ⅱ．１～Ⅱ．５と後述したが、業務モデルやデータモデルに不案内の読者は、これらを先に読んでいただいたほうが良いかとも思われる。またこれら図法についての詳細は、筆者による下記の既刊１，２が参考になるかと思う。

既刊１：データ中心システムの概念データモデル（オーム社）

既刊２：名人椿正明が教えるデータモデリングの技（株式会社翔泳社）

したがって、本書の主張する方法論は次のように要約できると思われる。

１．適切な大きさの相互に独立して運用できるデータ標準化の文化圏、データハブ（ＤＨ）を前提にアプリケーションを開発する。ＤＨはデータ管理者によるデータ管理を効率化し、コミュニケーションを効率化するために設定する。

２．リソースデータ（通称マスターデータ）とイベントデータ（通称トランザクションデータ)、またメタデータは、それぞれ別ＤＨとする。

３．イベント系ＤＨ間のコミュニケーションは、先日付在庫によって行う（Ｉ．３参照)。

４．データに関するコミュニケーションは、実装独立データのすべてからなるＴＨデータモデル図をベースに、ＳＰＦ（Schema Process Flow）処理をトレースしながら行う。

（注）ＴＨ以外の多くのデータモデルはＲＤＢにストアするデータに限るため、加工データなどが除かれ、整合性管理に欠落を発生することがある。

５．ユーザニーズは、ＩＰＦチャート上に登場したＩＯ（画面・帳票）を素材として確認する。

６．ＤＨ、エンティティタイプ、属性、ＩＯ、ＩＯデータ、業務プロセスなど、またその相互関係は標準情報資源部品として、ユーザ企業責任で、リポジトリで管理する（ＩＲＭ指向)。

７．ＩＯの実装仕様の標準化のためにＩＯ区画を導入し、実装仕様をＩＯ区画部品として記述し、プログラミングレスを実現する。

８．システムの実装・運用は、リポジトリを介して、ＳＩｅｒが担当する。

なおやや古い（１８年前）が、ＤＯＡの基本コンセプトや適用実績を紹介した既刊３の概要については、次の（かこみ）を参照されたい。

既刊３：データ中心システム入門（オーム社）

＊　（かこみ：データ中心システム入門（１９９４）からの抜粋）＊＊＊＊＊＊＊＊＊＊＊＊＊

・企業横断的データの見方ができるか。

・コミュニケーション、コンセンサスと言う人間にとって永遠の課題。

・ＰＯＡは対象アプリケーション内のデータの流通は考えるが、これを超えるものは個別アプリケーションやユーザ（人間系）に任せようとする。
・「情報はユーザの求めている製品であり、データはその素材・原料」
・システム規模とは何か、１回の開発の単位ではない。データの共用あるいは流通範囲である。
・すべての工業製品は標準部品共用の上に成り立っている。ところが情報システムの世界では、製品設計、すなわち画面・帳票を設計する人の頭の中に、何が標準部品で、何が非標準部品か、はっきり登録されていない場合が多い。
・データ項目は人、物、金に次ぐ第４の会社の資源などと言われているが、実態は会社のデータではなく、担当個人のデータとなっている。
・葉だけ見えて、枝も木も森も見えない状態では、活用のための管理はできない。
・流通のためには変換テーブルが必要になる。対応が１対１ならよいが、ｎ対ｍになっていると大変厄介なことになる。
・データ項目と定義域の違いは分子と原子の違いに相当する。データモデルの教育を受けていない人の８割はこの両者を混同している。
・データモデルは、データ部品を正しく客観的に識別するために不可欠である。データモデルがないとフィーリングでデータ部品が識別される。調整に時間がかかり声の大きい人が勝つ。
・建築、電気製品、化学プラントには基礎科学があるが、情報システムの世界には、それらしき基礎科学はなく・・・
・ＰＯＡの発想「概要から詳細へ」が残っていて、当初から詳細（全属性）を考える習慣がなく、エンティティタイプ＋主要属性にとどまることが多い。
・物理ＤＢの構造はほとんど見えている。いまさら概念レベルといわれてもかえってわかりにくい。
・概念レベルへの取り組みの遅れる最大の原因は、従来の概念と実装の渾然とした設計・開発方法論が身体にしみついているためである。
・データモデル図は一つの図面言語である。人は一つ言語を覚えるとそれにこだわる。したがって、はじめにできるだけ欠陥の少ないものを選びたい。
・戦略的に当初から効率を求めるのは誤りである。はじめは品質を求める。高品質をつくれる人の育成をねらう。効率はそれから求めればよい。
・データが概念レベルで表現できる以上、その変換の過程も概念レベルで表現できるはずだと私たちは考える。その表現手段として私たちはＳＰＦチャートを用いる。
・都市計画でも、実際の工事に先立って測量を行い現状を正しく把握する。地図づくりに際し、決して目的を一つにしぼったり、工事との対応を厳しく制御することはない。工事は第１期、第２期、第３期と分けるにしても、一括測量しておいて何ら不都合はないのである。現状調査と開発の分離をもっと考えるべきである。

・身体で覚えるものだけに水泳、ピアノ、英語、自動車の運転と同様、訓練が必須である。
・下流での生産性向上のためには上流の品質向上が有効である。
・　ＤＯＡプロジェクトの特徴は、回を重ねるごとに規模を小さくできることである。逆に言うとはじめが大変である。

＊＊

＊＊

おわりに

今回、２０１３年、サイロフリーおよびプログラミングレスの方法論を提案することになったが、その発端は、１９６５年頃、千代田化工建設(株)において、FORTRAN による、化学プロセスシミュレータを開発したことに遡る。いくつもの SUBROUTINE を用意し、これを組み合わせてシミュレータを作った際、SUBROUTINE をつなぐのに、ARGUMENT 方式と COMMON 方式のどちらをどう使うべきかが問題になった。本書では、前者を PUSH 型通信、後者を PULL 型通信と言って分類したが、そのときこの問題に初めて遭遇したことになる。

結局、「対象とするアプリケーションの実態を素直に反映した COMMON の設計」が鍵であり、全ての SUBROUTINE はこれを共有すべきこと、またこれを外部記憶に展開することによって、シミュレータに続く設計、調達、工事などの多くのアプリケーションとのコミュニケーションが、スムーズに展開できることに気づいた。そこで方法論としてＴＨデータモデルを、またツールとしてＤＤ／Ｄ――今の用語ではメタデータリポジトリ――内蔵型の一種のエンジニアリング DBMS「DPLS（Database Programbase Languagebase Support)」を開発し、１９７５年第１回ＶＬＤＢ（Very Large Data Base）国際学会に発表することになった。P. P. Chen がＥＲモデルを発表したのと同時である。

今振り返ると、「COMMON ベースのシミュレータ」は、外部記憶は持たなかったが、アーキテクチャとしては本書で言う世代２であり、次の「DPLS」ではリソースデータの一元化を提案していたので、ＭＤＭを実現する世代３のシステムアーキテクチャを提案していたことになる。また、２０００年ころであるが、個別アプリ間の通信場として、EDW（Enterprise Data Warehouse）を提案した。これを今回は世代４のＥ２データハブ（DH）と改称し、さらに企業間に通信場を設定する世代５のＥ３データハブ（DH）を提案することになった。

今回はまた、「PULL 型通信では、イベントレコードは先日付在庫に変換すれば、通信の目的が達成できる。したがって、イベントレコードをそのまま受信側に送信する必要はない」、また「Ｅ２やＥ３で扱うイベントレコードは、商品、サービス、情報／KH、金など財に限られるので、残高は相手の要求、いわば外圧にどれだけ対応しているかを示し、Ｍ２は、売上や生産など、自社の計画、いわば内圧にどれだけ対応しているかを示すものである」、ということを悟った。また「ニーズの進展とともに、通信は PUSH から PULL に進化し、データハブが１階層アップする」という仮説を確認した。

「対象とするアプリケーションの実態を素直に反映した COMMON の設計」は、ＴＨデー

タモデルを生み、また、アプリケーションの仕様は、データモデルおよび処理モデルとも、メタデータで書けるから、これをメタデータリポジトリに記述すれば、プログラム形式の記述の必要はないはずだ、と考えた。今回この仮説に対しては、これまでの概念ＤＢ設計のための画面・帳票のＩＯデータモデルには、実装にかかわる機械と人間の連携仕様やレイアウト設定などに不足があったので、ＩＯ区画（ＩＯＰ）やＩＯＰ要素を導入して、これを補うことになった。

１９７５年から２０１１年にわたる筆者引退前のＤＯＡは、サイロを作り後追いでこれらを繋ぎ、ビジネスや実装ニーズの変更には、プログラムを書いて対応する、依然メンテ地獄から開放されない、いわばＤＯＡ１．０というものであった。「これではＤＯＡの終点に到達できない、ＤＯＡ２．０を目指すべき」と、引退後ではあったが、３階層通信場モデルとプログラミングレスの方法論を提案することになった。開発力を持たないので、理論レベルにとどまっているが、本来あるべきシステム開発方法論に近づけたのではないかと思う。

ユーザインターフェースについては、まだ種々進化しつつあり、経験の少ない、筆者の提案には修正を要する点も――特に人間と機械の協業のパターン化にかかわる連携関数に関して――多々あるかと思われるが、データ中心アプローチの究極の姿が、ほぼ明らかになったのではないかと考える。

今後の課題は、迅速に適切なＥ１を識別し、その間にやり取りされるメッセージから、Ｅ２、Ｅ３を、さらに経営目標からＭ２の骨格を、効率的に可視化・共有する方法論を確立することであると考える。そして、各ユーザ企業においては、３～１０名の、データモデル図、業務処理モデル図を読みこなし、Ｅ１～Ｅ３を分析するＤＯＡ人材の育成が不可欠だと考える。こうしてユーザ企業の用意したブレのないビジネスの可視化ドキュメントを読みこなし、ベンダーが最新の情報技術を駆使してシステムの実装・維持・運用を担当する、そのような日が、一日も早く来るのを待ちたい。

ERP は実装仕様まで含めた形で、テンプレートを用意し、合理化しようというものであったが、ビジネス仕様と実装仕様の変化の組み合わせに対応しなければならず、メンテ負荷は大きく、またユーザとビジネスモデルを共有するには複雑すぎた。筆者の提案する 6 業種には不備があるとしても、業種には限りがあり、これ対応に実装独立ビジネスモデルのメタデータテンプレートを準備し、これをベースに企業ニーズや実装ニーズに対応するのが今後の方向かと思われる。データ総研には、この方向でのリーダーシップを期待する。